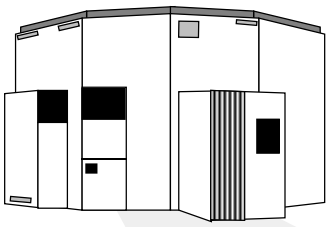
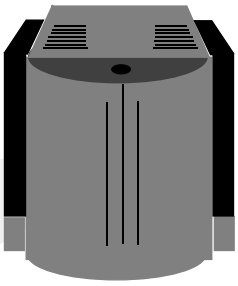
Naval Research Laboratory

Washington, DC 20375-5320





Publishing on the Web

Course Notes

Instructor: Michael G. Vonk
Center for Computational Science
(202)767-3884
michael.vonk@nrl.navy.mil

Publishing on the Web

1. Introduction

This course guides the student through the steps taken in providing information on the World Wide Web. Topics covered include:

- Web browsers
- Web server installation and configuration
- writing HyperText Markup Language (HTML) files
- using the Common Gateway Interface (CGI)
- generating pages on the fly
- developing forms
- creating image maps
- protecting Web pages

Examples for this course were tested on the NCSA httpd 1.4 and MacHTTP 2.2 Web servers. Other Web servers are available, but may support various features in server-specific ways.

This class provides an overview to the topics covered. References to additional, more detailed, information can be found at:

```
http://amp.nrl.navy.mil/code5595/
  ccs-training/web-publishing/
  companion-page.html
```

2. Web Browsers

Several Web browsers exist, the most common being Netscape (for which NRL has a site license) and Mosaic (public domain). Browsers are available for most operating systems, including:

- X Windows
- Macintosh
- Microsoft Windows

Depending on which browser is used, various features (such as forms and tables) may not work as expected. Lynx, a line-mode browser, is also available.

2.1. Navigating the Web

Navigating within a browser occurs in one of two ways:

- following links
- opening Uniform Resource Locators (URLs)

URLs have the following format:

scheme`://`*host*`.`*domain*`[:`*port*`]/`*path*`/`*filename*

where *scheme* is one of:

`http`	a file on a Web server
`ftp`	a file on an anonymous FTP server
`file`	a file on your local system
`news`	pointer to a Usenet newsgroup
`gopher`	a file on a Gopher server
`wais`	a file on a WAIS server

2.2. Document Types

Web browsers can display HTML and text files and certain types of images (GIF and bitmap image formats) directly. Other multimedia documents types, including images, animations, sound files, and PostScript documents, are handled via external programs, known as "viewers."

The browser first maps the file extension of the URL referenced to a particular MIME type. Then, based on the MIME type, the browser displays the file itself, or invokes an external program.

Common file extensions include:

`.html`	HTML document
`.txt` or `.text`	plain text file (default)
`.gif`	GIF format image file
`.ps`	PostScript file
`.xbm`	X-Bitmap (black & white) image file
`.xpm`	X-Pixelmap (color) image file
`.jpeg` or `.jpg`	JPEG-encoded image file
`.mpeg` or `.mpg`	MPEG-encoded movie file
`.au`	AIFF-encoded audio file
`.Z`	compressed file (UNIX `compress`)
`.gz`	compressed file (GNU `gzip`)

The external programs must be installed on your system in addition to the browser software itself.

3. Web Servers

Two possibilities exist for serving information on the Web:

- use space on Code 5595's Web server (contact Computational Support Services at 767-3884)
- install your own Web server on a UNIX workstation, Macintosh, PC, etc.

Installation of a Web server involves the following steps:

- downloading and compiling (if necessary) server software
- configuring the server (access control, resources used, etc.)
- starting the server (as needed or via system startup script)

A list of available public domain and commercial Web server software packages is available through the companion page for this class.

Network aliases should be set up (contact the Networking Group at 767-3903), so if your Web server is moved to a different machine, you simply change the alias, and users need not change their links.

A mail alias "`webmaster`" is commonly established so users can contact the server administrator easily.

4. HyperText Markup Language (HTML)

HTML documents are ordinary text files with embedded tags used for formatting and to reference other documents, images, etc. HTML tags are usually paired, but may appear singularly:

```
<tag>text</tag>
<tag attribute=value>text</tag>
<tag>
```

Tags may be interpreted differently depending on the browser used. For example, a first level header may appear in a certain font and size on one browser, but in another font and size on another browser. Tags are not case sensitive.

4.1. HTML Document Format

The basic format of an HTML document is as follows:

```
<html>
  <head>
    <title>Title text</title>
  </head>
  <body>
    Body Text
  </body>
</html>
```

Although shown indented here, formatting within HTML documents is ignored (except within preformatted text tags). Formatting of displayed text is controlled solely by the embedded markup tags.

HTML documents are comprised of two parts:

`<head>`	information about the document, such as its title, not displayed as part of document text
`<body>`	content of document, including formatting tags, and link references

4.2. Basic Markup Tags

4.2.1. Title

Used in head section (all other tags listed are used in body) and displayed in the browser's title bar and hotlist (bookmark) menu:

```
<title>...</title>
```

For example:

```
<title>Publishing on the Web</title>
```

4.2.2. Headers

Six levels of headers can be used:

```
<h1>...</h1>
<h2>...</h2>
      .
      .
      .
<h6>...</h6>
```

Level one headers are typically the same as the document title.

4.2.3. Paragraphs and Line Breaks

The paragraph tag <p> is used to terminate paragraphs. Paragraphs are separated with blank lines when displayed. If paragraph tags are not used, text will run together. Extra white space (spaces, tabs, and new lines) within paragraphs is ignored.

New lines can be generated (without the extra blank line associated with paragraph tags) using the line break tag
.

4.2.4. Preformatted Text

Preformatted text, in which spaces, tabs, and new lines are significant, is typically displayed in a fixed-width font:

```
<pre>
text
    .
    .
    .
</pre>
```

Preformatted text is especially useful for program listings and columnar data. Links can be used within preformatted text—other tags should be avoided.

4.2.5. Horizontal Rules

A horizontal line can be drawn using the <hr> tag.

4.2.6. Comments

Comments can be inserted into an HTML document using:

```
<!-- comment -->
```

4.2.7. Lists

Unordered (Bullet) Lists

```
<ul>
    <li>First item
    <li>Next item
</ul>
```

Ordered (Numbered) Lists

```
<ol>
    <li>First item
    <li>Next item
</ol>
```

Definition (Glossary) Lists

```
<dl>
    <dt>First term
    <dd>Definition of first term

    <dt>Next term
    <dd>Next definition
</dl>
```

4.2.8. Character Emphasis

Two types of character emphasis tags are available:

- Logical Styles *type* of emphasis specified
- Physical Styles *appearance* of text specified

Logical Styles

	Emphasis
	Stronger emphasis
<code>	Computer code
<samp>	Sample output
<kbd>	Text to be entered from a keyboard
<var>	Variable text to be entered from a keyboard
<dfn>	Display a definition
<cite>	Display a citation

Physical Styles

<b>	Bold font
<i>	Italics
<u>	Underline
<tt>	Typewriter font

4.2.9. Special Characters

Certain characters have special meaning to HTML, such as <, >, &, and ". To express these characters, use the following:

```
&keyword;
```

For example:

```
&lt;
&gt;
&amp;
&quot;
```

References to the ISO Latin-1 character set can also be used. In addition, you can specify a character's numeric value:

```
&#ascii_equivalent;
```

4.3. Inline Images

Images (in either GIF or bitmap format) can be displayed inline:

```
<img src="URL">
```

Users not using a graphical interface, such as Lynx users, or who have turned off image loading, will see the message "[IMAGE]" where the image would normally appear. To specify alternative text, use:

```
<img src="URL" alt="[imagetitle]">
```

Images may be placed on separate lines, or aligned with adjoining text using the ALIGN attribute with values BOTTOM (default), MIDDLE, or TOP.

Displaying large images can take a long time. It is better to create a separate "snapshot" version of a large image, and let the user click on this snapshot to display the larger image.

4.4. Linking to Other Documents

Hypertext links are specified using the anchor tag. Links can be made from textual phrases or from images to any URL. Links are specified as follows:

```
<a href="URL">...</a>
```

For example:

```
This is a link to the <a href="http://
amp.nrl.navy.mil/code5595/ccs-training">
CCS Training</a> page.<p>
```

Links can be made from images, as follows:

```
<a href="large-image.jpeg">
    <img src="snapshot.gif">
</a>
```

URLs can be specified three ways:

- Fully specified URLs
- Absolute path names (starting at Web server root)
- Relative path names (starting from current document)

Depending of the type of file referenced, the browser may display the document in its own window, or invoke an external program to handle the request.

4.4.1. Linking to Locations Within Text Documents

Links, by default, take you to the beginning of the specified HTML document. To define a location for linking within a document, it must first be named:

```
<a name="target">...</a>
```

You can then link directly to the named target within another or the same document:

```
<a href="URL#target">...</a>
<a href="#target">...</a>
```

4.5. HTML Editors and Converters

HTML documents can be created using any editor or word processor (save as plain ASCII text) , including:

- vi
- emacs
- Microsoft Word
- Word Perfect

Numerous WYSIWYG HTML editors are available, as well as utilities for converting existing document formats into HTML.

5. Common Gateway Interface

The Common Gateway Interface (CGI) is used to invoke external programs (or "gateways") in order to:

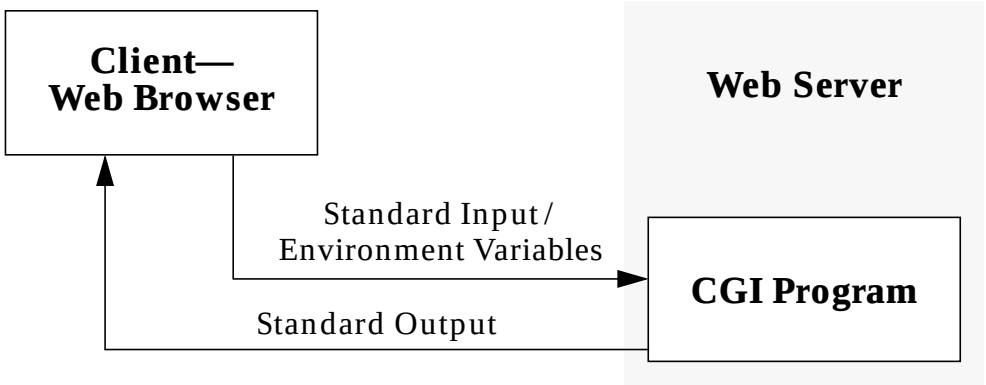
- generate Web pages on the fly
- handle image map requests
- process form data

CGI programs can be written in any language which produces an executable file, including:

- Perl
- C / C++
- Shell scripts

5.1. Invoking a Gateway Program

When a client (browser) requests the URL corresponding to a gateway program, the server executes the program and sends output back to the client.



5.2. Passing Data to a CGI Program

Data is passed to the gateway program via standard input or via environment variables, including:

`QUERY_STRING`	text following "?" in the URL
`PATH_INFO`	Extra information appended to the URL

Forms use standard input to pass form results to form handler programs. Image maps pass map configuration files via `PATH_INFO`. Data can also be hardcoded in URL references.

5.3. Returning Information to the Client

Data (either an actual document or a reference to another document) is passed from the gateway program back to the client via standard output. Since file extensions cannot be used to determine document type for automatically generated documents, you must include this information.

To return an actual document, the following header is used:

`Content-type:` *`type/subtype`*

where *`type/subtype`* is the MIME document type. Common MIME types include:

- `text/html`
- `text/plain`

The content of the document is then written to standard output separated from the content type header by a blank line. For example:

```
print "Content-type: text/html\n";
print "\n";
print "<html>\n";
print "<head>\n";
print "  <title>Script Output</title>\n";
print "</head>\n";
print "<body>\n";
print "<h1>Script Output</h1>\n";
print "Hello, world...<p>\n";
print "</body>\n";
print "</html>\n";
```

To return a reference to another document, use the following:

```
Content-type:  type/subtype
Location: URL
```

6. Forms

Creating forms involves two steps:

- writing an HTML form specification
- writing a script to handle the results of the form

Form results are passed to the form handler script via standard input.

6.1. Form Creation

Forms can be placed anywhere within the body of an HTML file and are defined as follows:

```
<form method="method" action="script-name">
        .
        .
        .
    form body: including prompting text,
               field definitions, and other
               html tags
        .
        .
        .
</form>
```

The *method* can be either `POST` (the preferred method) or `GET`. The value of the `action` attribute is the URL of the script that will be invoked to handle form results.

Forms are not visually distinguished within HTML documents—horizontal rules are often used to delimit form boundaries.

6.2. The `INPUT` Tag

The `INPUT` tag is used to specify a simple input element. Attributes of an `INPUT` tag include:

- `TYPE` must be one of:
  "`text`"
  "`password`"
  "`checkbox`"
  "`radio`"
  "`submit`"
  "`reset`"
- `NAME` defines input field name (required for all except "`submit`" and "`reset`" types)
- `VALUE` depends on field type:
  for "`text`" or "`password`", specifies default value to be displayed
  for "`checkbox`" or "`radio`", specifies value of button when checked (default is "on")
  for "`submit`" or "`reset`", specifies label for pushbutton
- `CHECKED` for "`checkbox`" or "`radio`", specifies that field is checked by default
- `SIZE` for "`text`" and "`password`", specifies field length in characters (default is 20)
- `MAXLENGTH` for "`text`" and "`password`", specifies maximum number of characters accepted

6.2.1. Text and Password Fields

Text fields are used to enter single lines of text:

```
Enter name: <input type="text" name="name">
```

In the form handler, text entered by the user is assigned to the variable specified with the name attribute. Password fields are simply text fields where input text is displayed on the screen using special characters.

6.2.2. Checkbox and Radio Fields

Checkboxes allow the user to select zero or more options from a list:

```
Select some numbers:<br>
<input type=checkbox name=a value=1> 1<br>
<input type=checkbox name=a value=2> 2<br>
<input type=checkbox name=a value=3> 3<p>
```

Radio boxes allow the user to select one option:

```
Select a number:<br>
<input type=radio name=b value=1> 1<br>
<input type=radio name=b value=2> 2<br>
```

6.2.3. Submit and Reset Fields

When the user clicks on a submit button, the form handler script named by the action attribute is invoked. The reset button resets all fields to their default values:

```
<input type=submit value="Submit">
<input type=reset  value="Reset Form">
```

6.3. The SELECT Tag

The SELECT tag allows users to select from a list of options. For example:

```
<SELECT NAME="options">
    <OPTION>Option 1
    <OPTION>Option 2
    <OPTION SELECTED>Option 3
</SELECT>
```

Attibutes to SELECT include:

- NAME — symbolic name of SELECT element
- SIZE — specifies how many items are visible at one time
- MULTIPLE — if present, allows multiple options to be selected

Attibutes to OPTION include:

- SELECTED — specifies that option is selected by default

6.4. The TEXTAREA Tag

The TEXTAREA tag is used for multiline text entry. Attributes include:

- NAME symbolic name of TEXTAREA element
- ROWS number of rows displayed
- COLS number of columns displayed

For example:

```
<TEXTAREA NAME="textarea1" ROWS=4 COLS=40>
default contents
</TEXTAREA>
```

The closing tag is required even if no default contents are specified. TEXTAREAs are displayed with scrollbars. Any amount of text can be input.

6.5. Handling Form Results

When the user clicks on the submit button, a list of field names and their associated values is passed, via standard input, to the form handler program specified in the ACTION attribute.

Special characters are encoded. The form handler program is responsible for decoding them.

7. Image Maps

Images can be used as maps to information on the Web. Regions of an image can be defined so that if a user clicks within a region, a specified action will be performed.

Steps in making an image map:

- creating the image
- creating an image map configuration file
- referencing the image map

7.1. Creating Images

Many image editing programs exist, such as `xpaint`, for creating images. The only requirement being that the program must be able to save images in GIF or bitmap format.

You must also be able to determine the coordinates of regions within the image. This can be accomplished using programs such as `xv` or `mapedit`.

7.2. Image Map Configuration Files

The image map configuration file is used to define regions within an image and the URL's they reference. Regions are described as follows:

```
method url coordinate-list
```

where *`method`* is one of the following:

`default`	specifies default URL
`circle`	takes two coordinates—center and edgepoint

`poly`	polygon of at most 100 vertices—each coordinate is vertex
`rect`	takes two coordinates—upper-left and lower-right
`point`	for closest to a point, takes one coordinate

Methods are evaluated in the order specified—this is important when regions overlap. Also, note that it does not make sense to specify a `default` when `point` is also used.

7.3. Referencing Image Maps

Image maps are handled by the `imagemap` program (typically found in the server's `cgi-bin` directory), and are referenced as follows:

```
<A HREF=
  "/cgi-bin/imagemap/directory/mapfile">
  <IMG SRC="map" ISMAP>
</A>
```

This will invoke `imagemap` and pass to it the name of the map file contained in */directory/mapfile* (via `PATH_INFO` as extra information).

You must specify the full path to your map file, starting at the Web server's root directory. For example:

```
<a href=
  "/cgi-bin/imagemap/code5595/ccs-training/
    web-publishing/image-maps/example.map">
  <img src="example.gif" ISMAP>
</a>
```

8. Access Control and User Authentication

Access control can be based on the following criteria:

- Usernames / passwords
- Internet addresses

There is no correspondence between httpd usernames and passwords and those on the UNIX system.

There are two levels at which authentication can work:

- Per server
- Per directory

8.1. Per Server Access

Global access configuration is controlled in `access.conf`.

8.2. Per Directory Access

The following steps are taken to control access to a particular directory:

- Create `.htaccess` file in directory
- Create or update the `.htpasswd` file (in another directory)
- Create or update the `.htgroup` file (also in another directory) if using access groups

8.3. Security Considerations

Passwords are passed over the networked uuencoded. This is roughly as safe as `telnet` style username / password security.

9. Document Style

There are several points that should be considered when publishing information of the Web:

- Signatures and Timestamps

  Include author's (webmaster's) email address, date of last modification, and disclaimer if necessary

- Context

  Text, titles, and vocabulary should stand by themselves—you can't rely on users having followed any particular path to a page

- Balance

  User's don't want to read through long text passages—keep message simple, break up long documents, use lists, and plenty of white space

- Consistency

  All documents on your server should have a consistent look and feel—logos at the top of each page and navigation icons at the bottom allow users to quickly sense where they are and move easily within your server

- Graphics

  Large images take time and should be used sparingly

- Advanced Features

  Some browsers implement non-standard features—use standard HTML

10. Summary

The World Wide Web and the browser programs used to access Web servers are perhaps the fasting growing computer applications ever.

Setting up your own Web server is relatively easy. Publishing information on the Web gives you global visibility, allowing you to access a potential audience of tens of millions of people. Maintaining and updating Web pages is straight forward and cost effective.

Two points to consider are:

- Information should be kept up to date. Unlike publishing a book, where the reader assumes that some of the information may have changed since it was printed, information on the Web can and should be updated constantly.

- When publishing information on the Web, or on any other information server, you are representing NRL. Personal opinions and information should be stated as such. The NRL publications office is in the process of establishing instructions for what and how NRL information will be disseminated via the Web.

Hopefully these notes will help you effectively publish information on the Web.